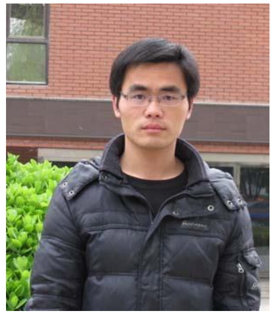
Flaky Test Detection in Android via Event Order Exploration



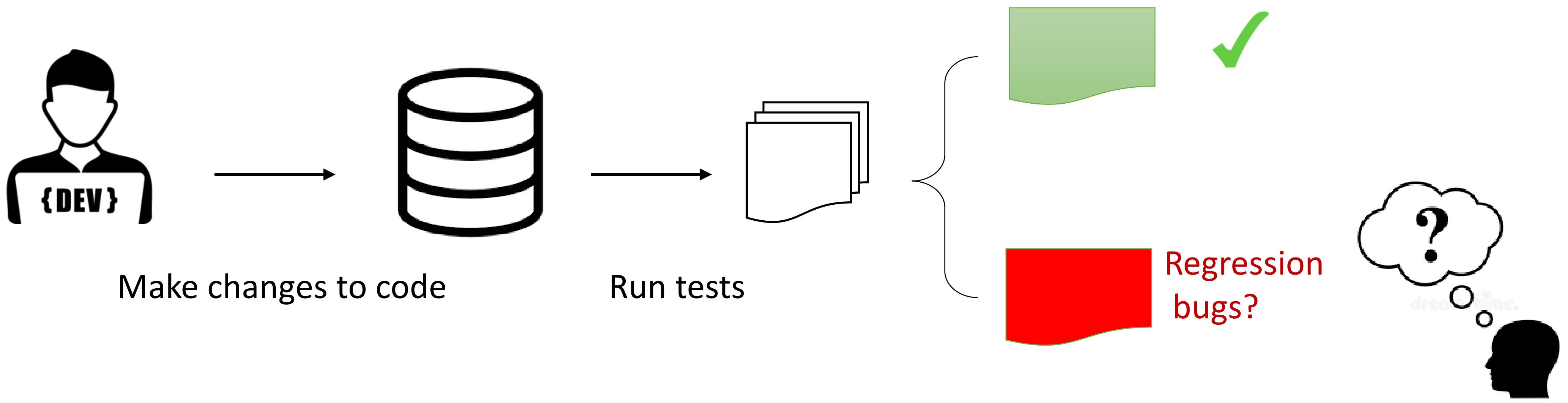
Zhen Dong Abhishek Tiwari Xiao Liang Yu Abhik Roychoudhury

Flaky Tests

- A test is considered ***flaky*** if it non-deterministically passes or fails when running on the same version of code[1]

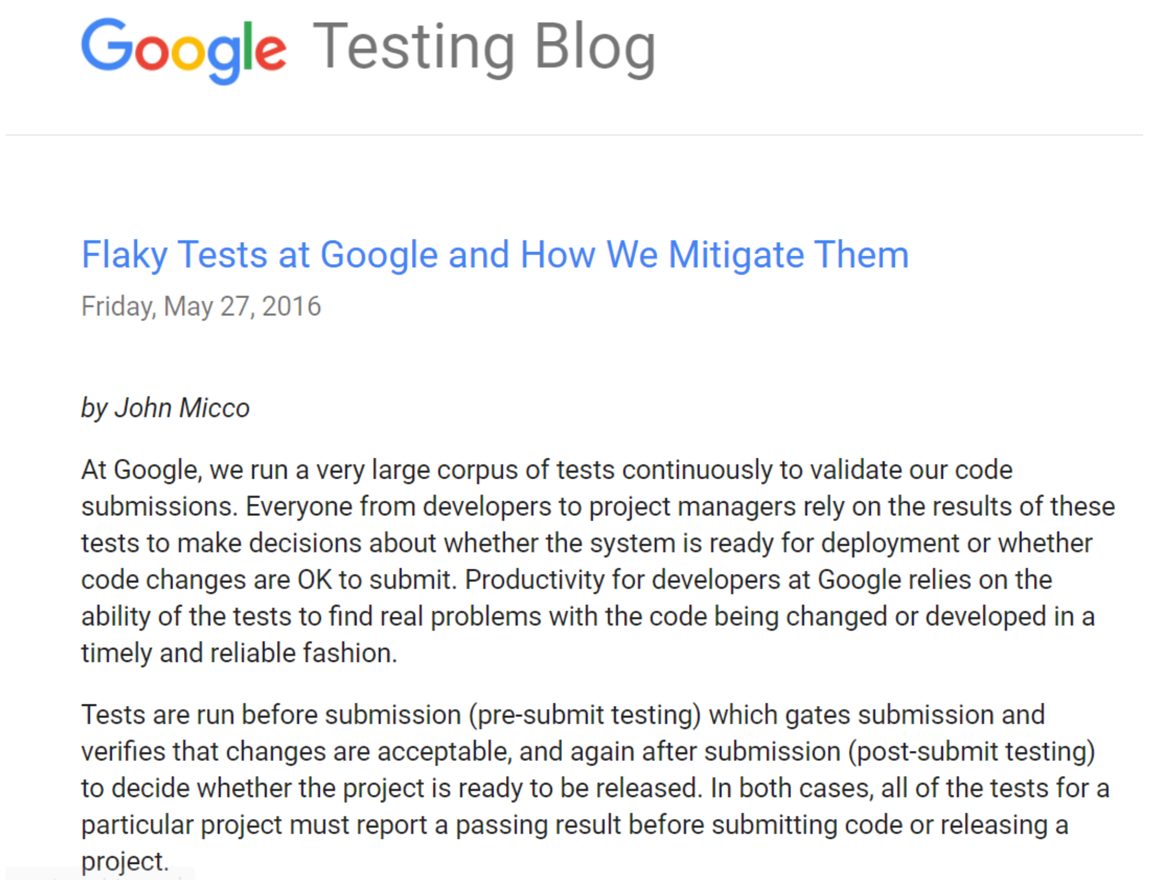
[1] J. Bell, O. Legunsen, M. Hilton, L. Eloussi, T. Yung, and D. Marinov. 2018. DeFlaker: Automatically Detecting Flaky Tests. In 2018 IEEE/ACM 40th International Conference on Software Engineering (ICSE)

Problems with Flaky Tests



- Reducing developer’ trust in test results
- Waste developer’s time on debugging nonexistent fault in code

Flaky Tests—A Real World War



Google Testing Blog

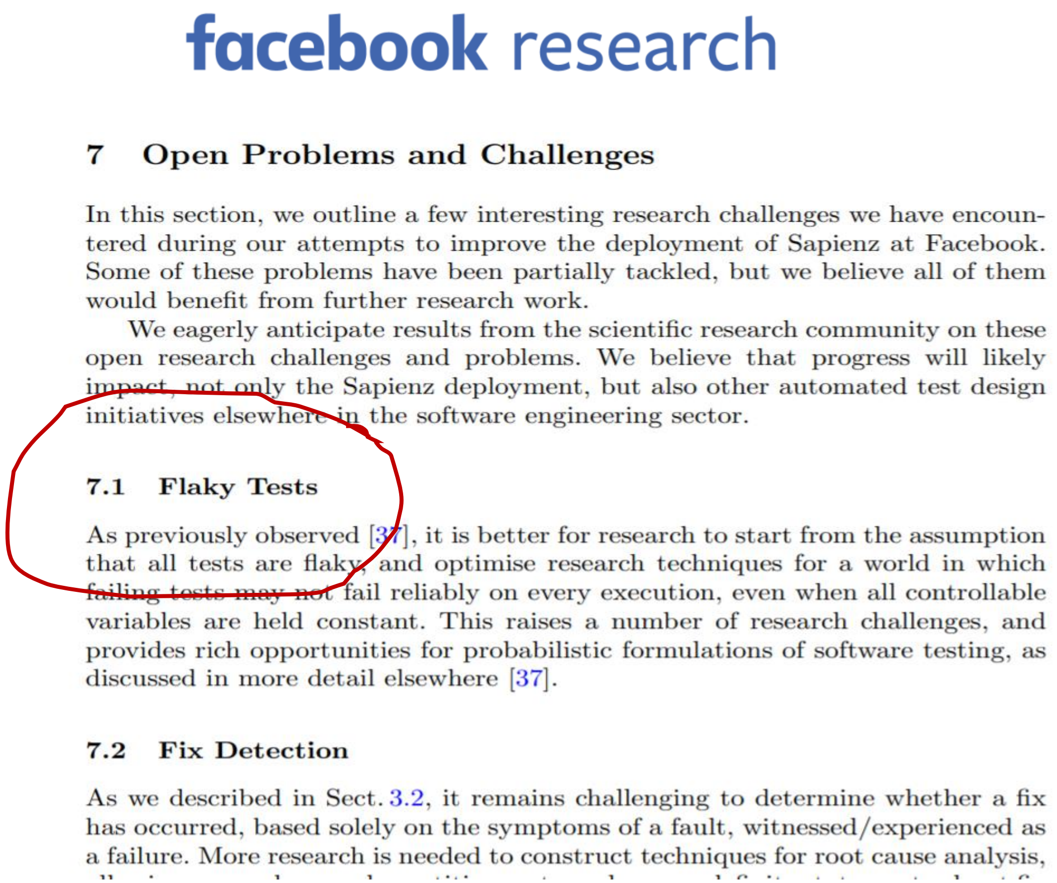
Flaky Tests at Google and How We Mitigate Them

Friday, May 27, 2016

by John Micco

At Google, we run a very large corpus of tests continuously to validate our code submissions. Everyone from developers to project managers rely on the results of these tests to make decisions about whether the system is ready for deployment or whether code changes are OK to submit. Productivity for developers at Google relies on the ability of the tests to find real problems with the code being changed or developed in a timely and reliable fashion.

Tests are run before submission (pre-submit testing) which gates submission and verifies that changes are acceptable, and again after submission (post-submit testing) to decide whether the project is ready to be released. In both cases, all of the tests for a particular project must report a passing result before submitting code or releasing a project.



facebook research

7 Open Problems and Challenges

In this section, we outline a few interesting research challenges we have encountered during our attempts to improve the deployment of Sapienz at Facebook. Some of these problems have been partially tackled, but we believe all of them would benefit from further research work.

We eagerly anticipate results from the scientific research community on these open research challenges and problems. We believe that progress will likely impact, not only the Sapienz deployment, but also other automated test design initiatives elsewhere in the software engineering sector.

7.1 Flaky Tests

As previously observed [37], it is better for research to start from the assumption that all tests are flaky, and optimise research techniques for a world in which failing tests may not fail reliably on every execution, even when all controllable variables are held constant. This raises a number of research challenges, and provides rich opportunities for probabilistic formulations of software testing, as discussed in more detail elsewhere [37].

7.2 Fix Detection

As we described in Sect. 3.2, it remains challenging to determine whether a fix has occurred, based solely on the symptoms of a fault, witnessed/experienced as a failure. More research is needed to construct techniques for root cause analysis,

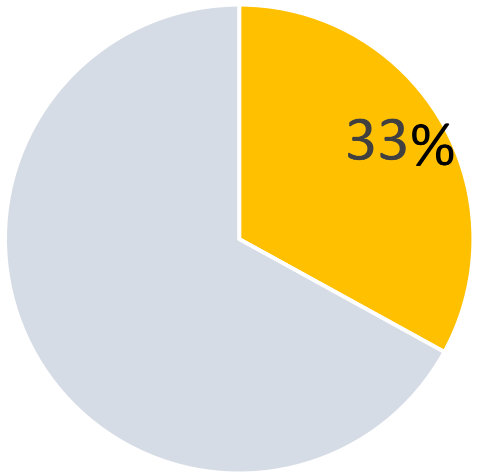
1.5% of test results are flaky

16% of our tests have some level of flakiness

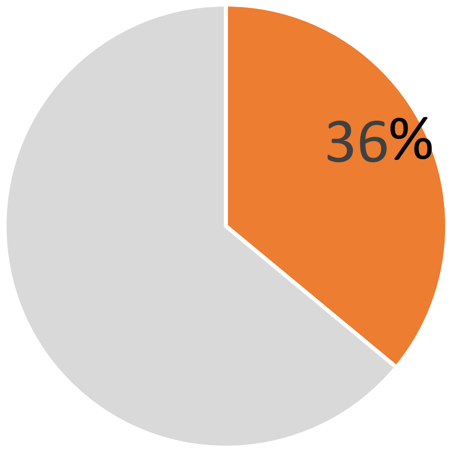
Assume All Tests Are Flaky

Detecting Concurrency-Related Flaky Tests in Android Apps

- Concurrency-related flaky tests are the most common flaky tests in Android

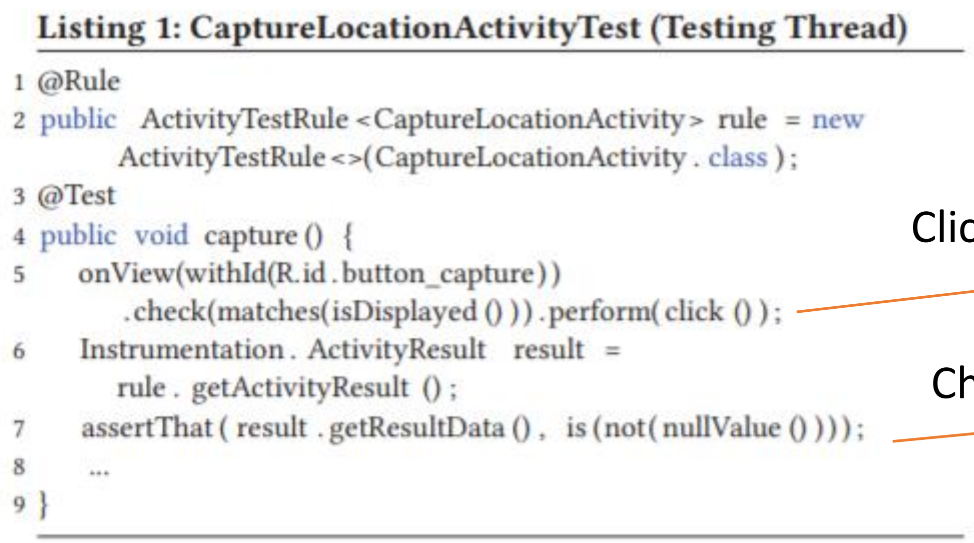


[1] Alan Romano, Zihe Song, Sampath Grandhi, Wei Yang, and Weihang Wang. 2021. An Empirical Analysis of UI-based Flaky Tests. In IEEE/ACM International Conference on Software Engineering



[2] Swapna Thorve, Chandani Sreshtha, and Na Meng. 2018. An Empirical Study of Flaky Tests in Android Apps. In International Conference on Software Maintenance and Evolution (ICSME).

An Example Flaky Test in Android Apps

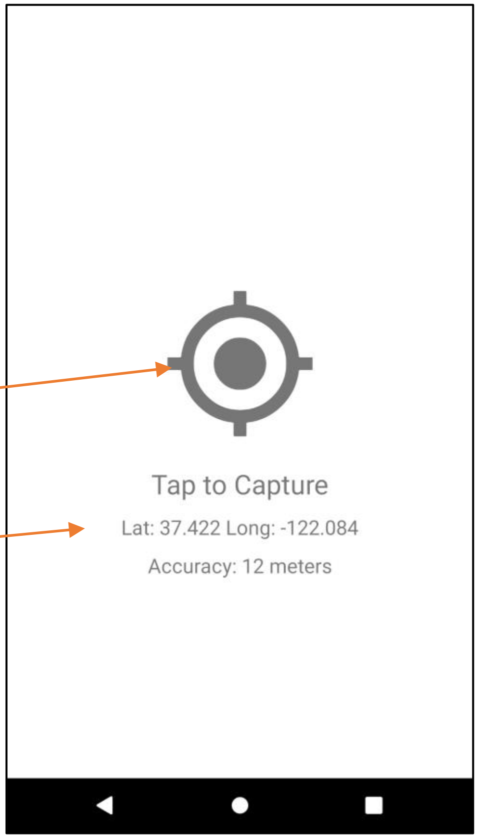


Listing 1: CaptureLocationActivityTest (Testing Thread)

```
@Rule
public  ActivityTestRule<CaptureLocationActivity> rule  = new
        ActivityTestRule<>(CaptureLocationActivity . class );
@Test
public  void  capture () {
    onView(withId(R.id . button_capture))
        .check(matches(isDisplayed () )).perform( click () );
    Instrumentation . ActivityResult   result  =
        rule . getActivityResult  () ;
    assertThat ( result . getResultData () ,  is (not( nullValue () )));
    ...
}
```

Clicking button

Checking results



RapidPro Surveyor

Passes in some runs and fails in others

An Example Flaky Test in Android Apps

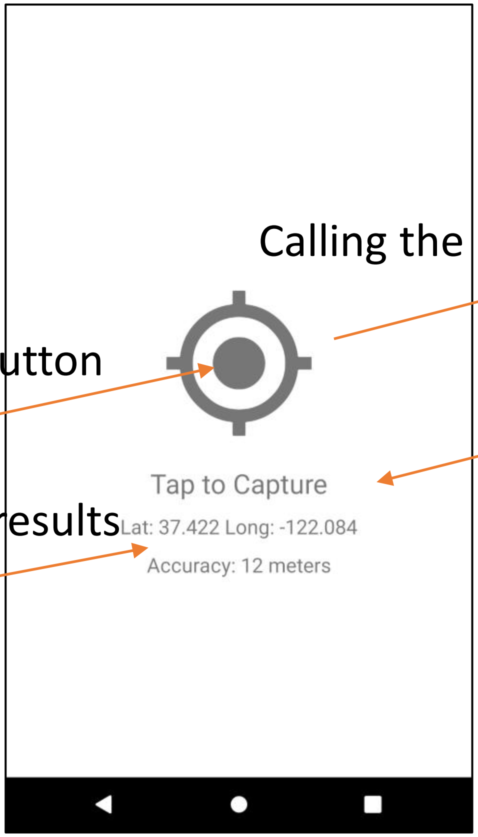
Pass

Listing 1: CaptureLocationActivityTest (Testing Thread)

```
@Rule
public  ActivityTestRule <CaptureLocationActivity> rule  = new
    ActivityTestRule<>(CaptureLocationActivity . class );
@Test
public  void  capture ()  {
   onView(withId(R.id . button_capture))
     .check(matches(isDisplayed () )) . perform( click () );
   Instrumentation . ActivityResult   result   =
      rule .  getActivityResult  () ;
   assertThat ( result  . getResultData () ,   is (not( nullValue () )));
   ...
}
```

Clicking button

Checking results



Calling the handler

Listing 2: CaptureLocationActivity (UI Thread)

```
@Override
protected  void  onCreate(Bundle bundle)  {
   connectGoogleApi();
    ...
}
protected  void  connectGoogleApi()  {
   googleApiClient   =  new  GoogleApiClient.Builder ( this )   ...
   googleApiClient . connect () ;
}
```

Updating results

Returning results

Launching an async task

Listing 3: Zaau (Worker Thread)

```
//Worker thread asynchronous to  the  UI  thread
@WorkerThread
public  void  run()  {
   zaak.zac( this . zagj ) . lock () ;
   ...
}
```

An Example Flaky Test in Android Apps

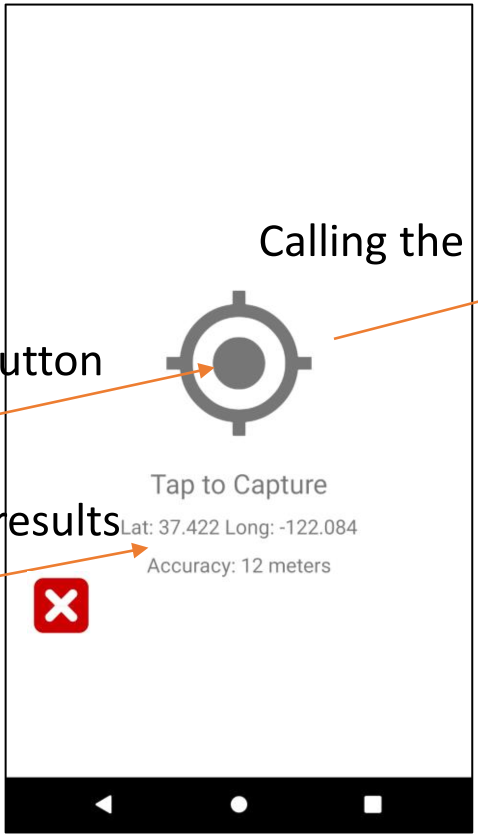
Fail

Listing 1: CaptureLocationActivityTest (Testing Thread)

```
@Rule
public  ActivityTestRule <CaptureLocationActivity> rule  = new
    ActivityTestRule<>(CaptureLocationActivity . class );
@Test
public  void  capture ()  {
   onView(withId(R.id . button_capture))
       .check(matches(isDisplayed () )) . perform( click () );
   Instrumentation . ActivityResult   result   =
       rule .  getActivityResult  () ;
   assertThat ( result  . getResultData () ,   is (not( nullValue () )));
   ...
}
```

Clicking button

Checking results



Calling the handler

Listing 2: CaptureLocationActivity (UI Thread)

```
@Override
protected  void  onCreate(Bundle bundle)  {
   connectGoogleApi();
    ...
}
protected  void  connectGoogleApi()  {
   googleApiClient  = new GoogleApiClient.Builder ( this )   ...
   googleApiClient . connect () ;
}
```

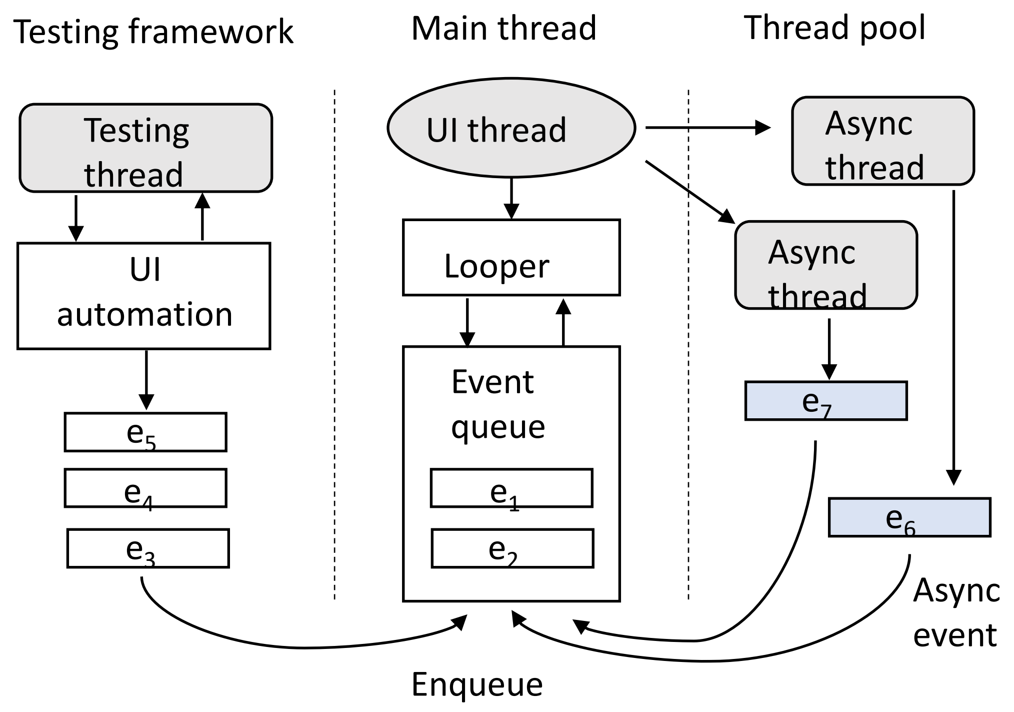
Launching an async task

Listing 3: Zaau (Worker Thread)

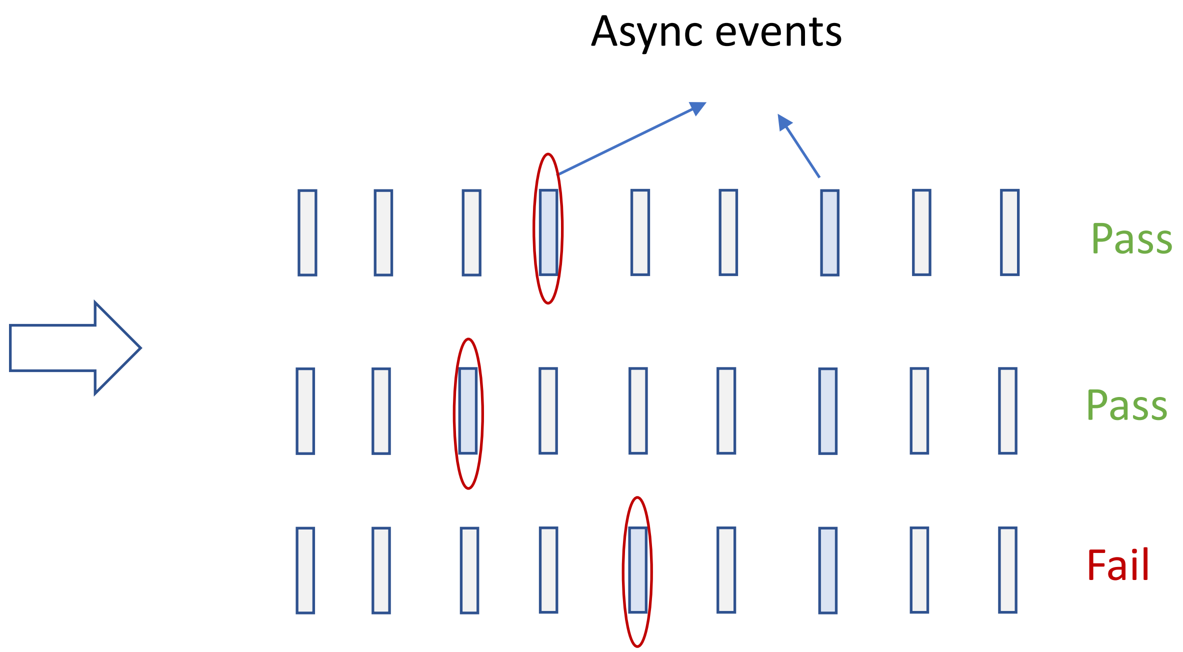
```
// Worker thread asynchronous to  the  UI  thread
@WorkerThread
public  void  run()  {
   zaak. zac( this . zagj ) . lock () ;
    ...
}
```

Taking longer to get results

Insight: Non-deterministic Execution of Async Events Causes Flaky Tests

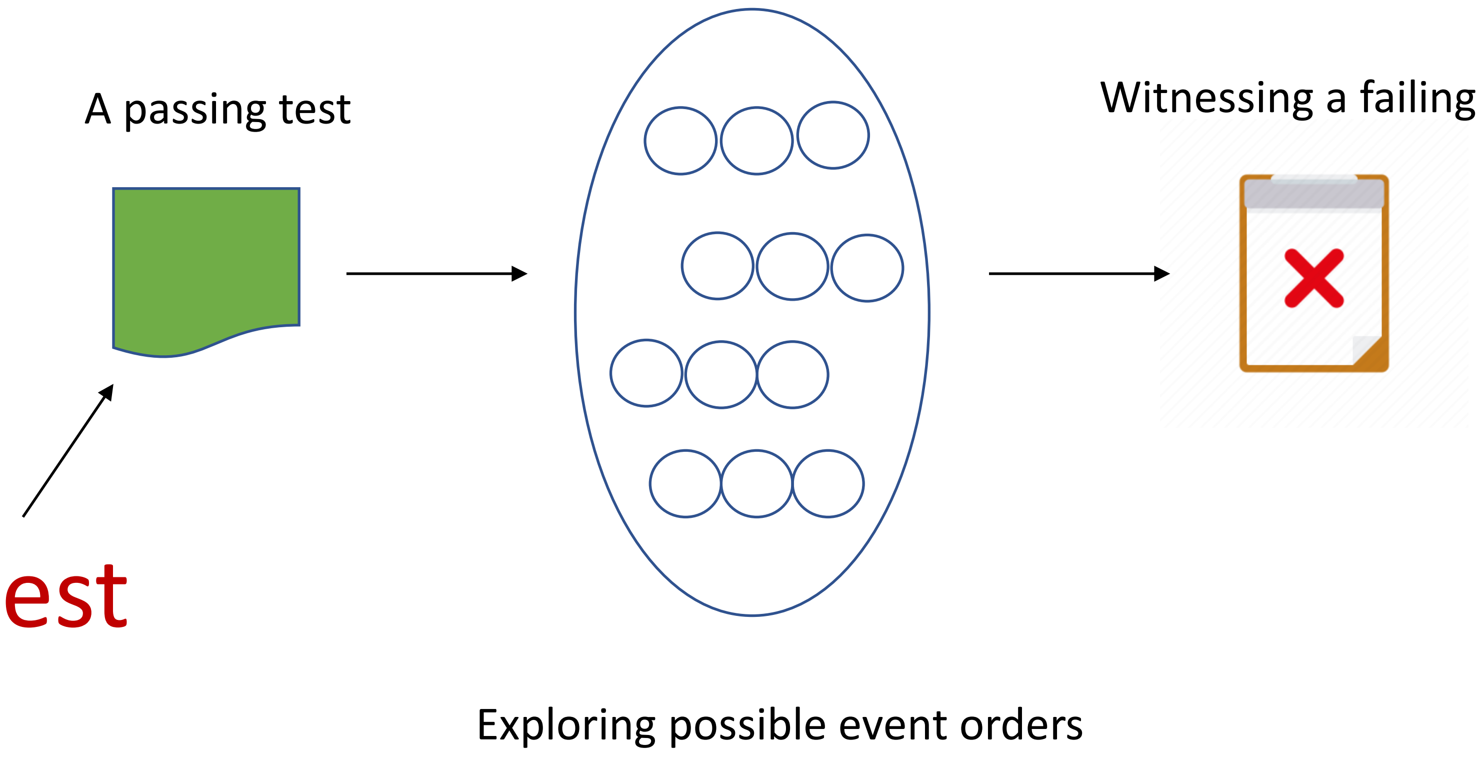


Android Event-Driven Concurrency Model



Possible event execution orders of a test

Idea: Detecting Flaky Tests by Exercising Different Event Execution Orders



Exploring Possible Event Execution Orders by Scheduling Async Events

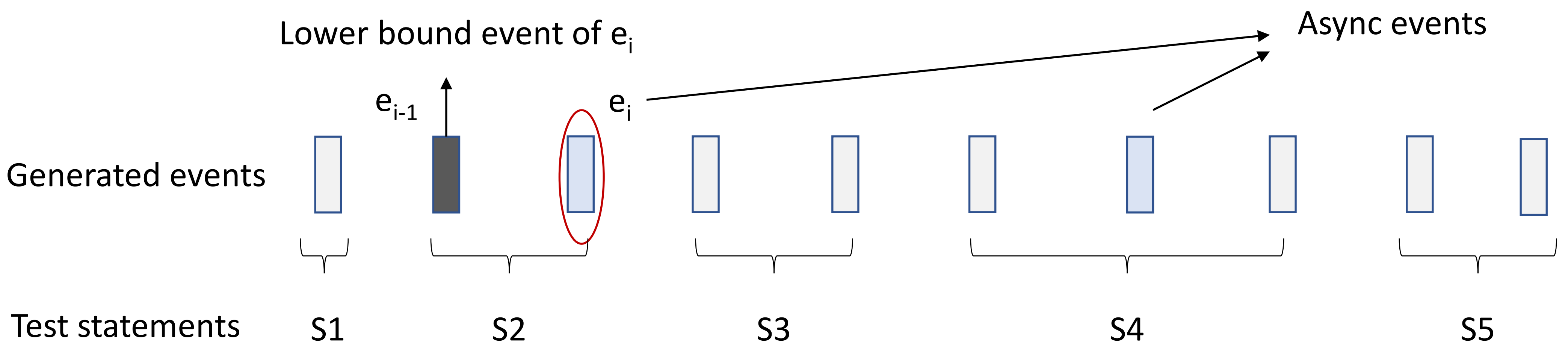
- Identify schedule space for an async event
- Scheduling an async event

Identifying Schedule Space of An Async Event

- Localizing its lower bound event
    - the latest event that the async event can not be executed earlier than.
- Localizing its upper bound event
    - the earlier event that the async event cannot be executed later than.
- Schedule space is between the bound event and upper event

Identifying Schedule Space via Dynamic Analysis

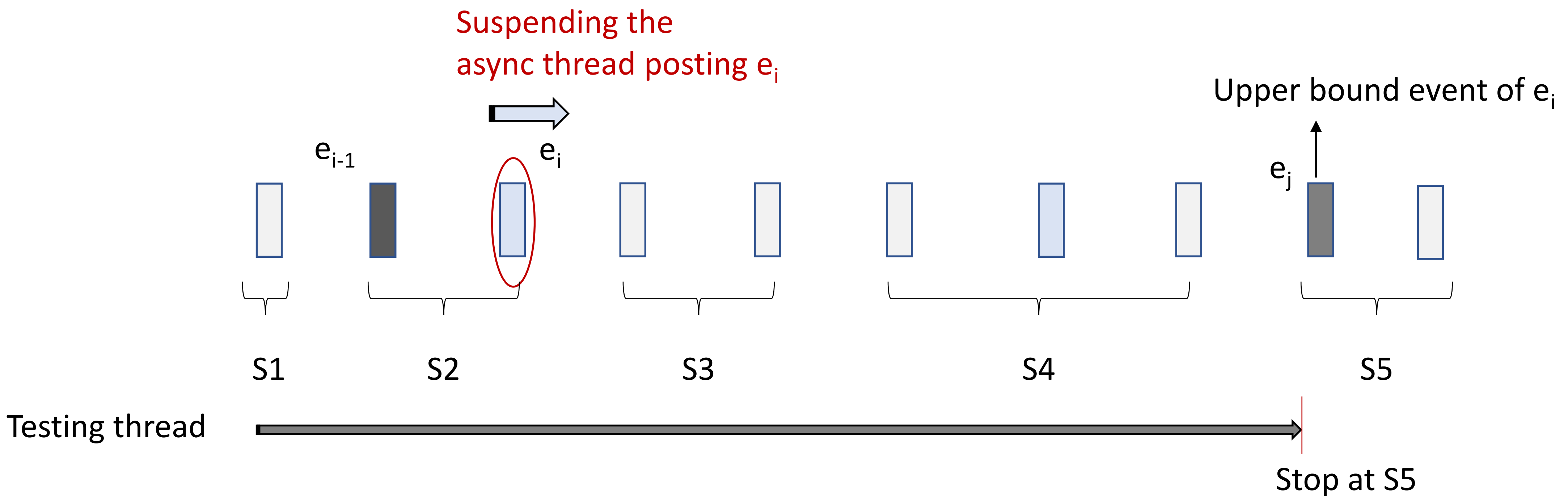
- Localizing the lower bound event



by mapping events to test statements

Identifying Schedule Space via Dynamic Analysis

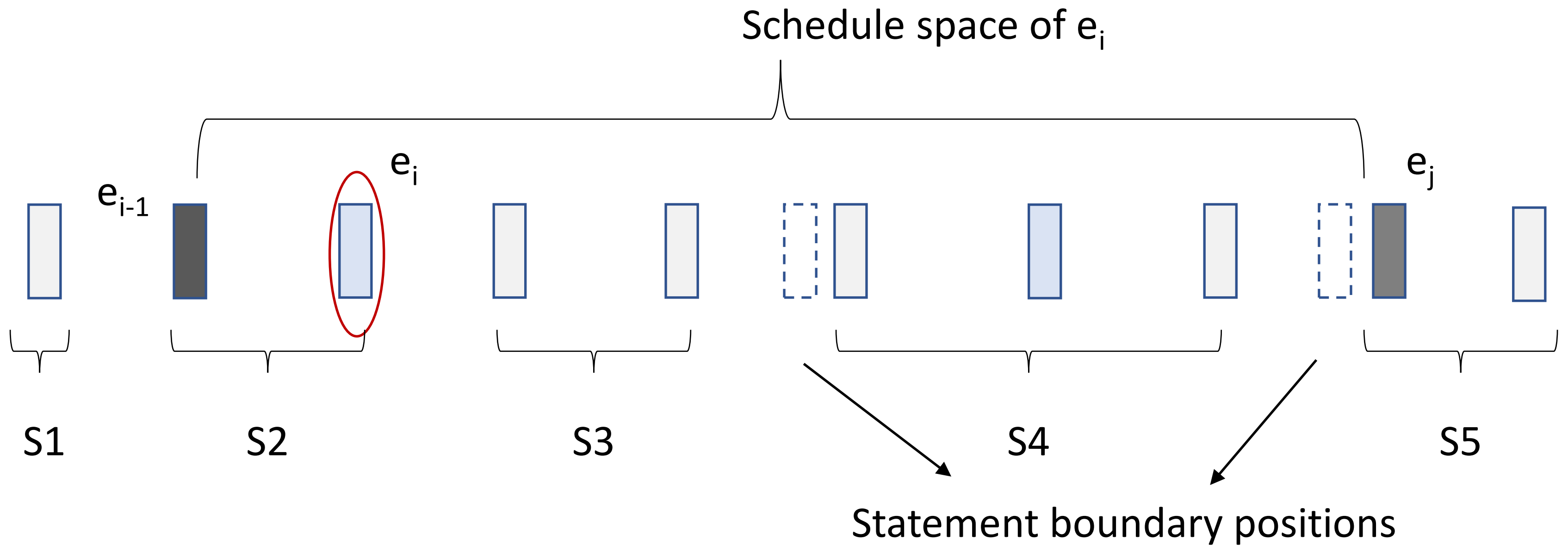
- Localizing upper bound event via thread operations


Suspending the async thread posting e_i
Upper bound event of e_i
e_{i-1}
e_i
e_j
S1
S2
S3
S4
S5
Testing thread
Stop at S5

Execution of S5 depends on e_i

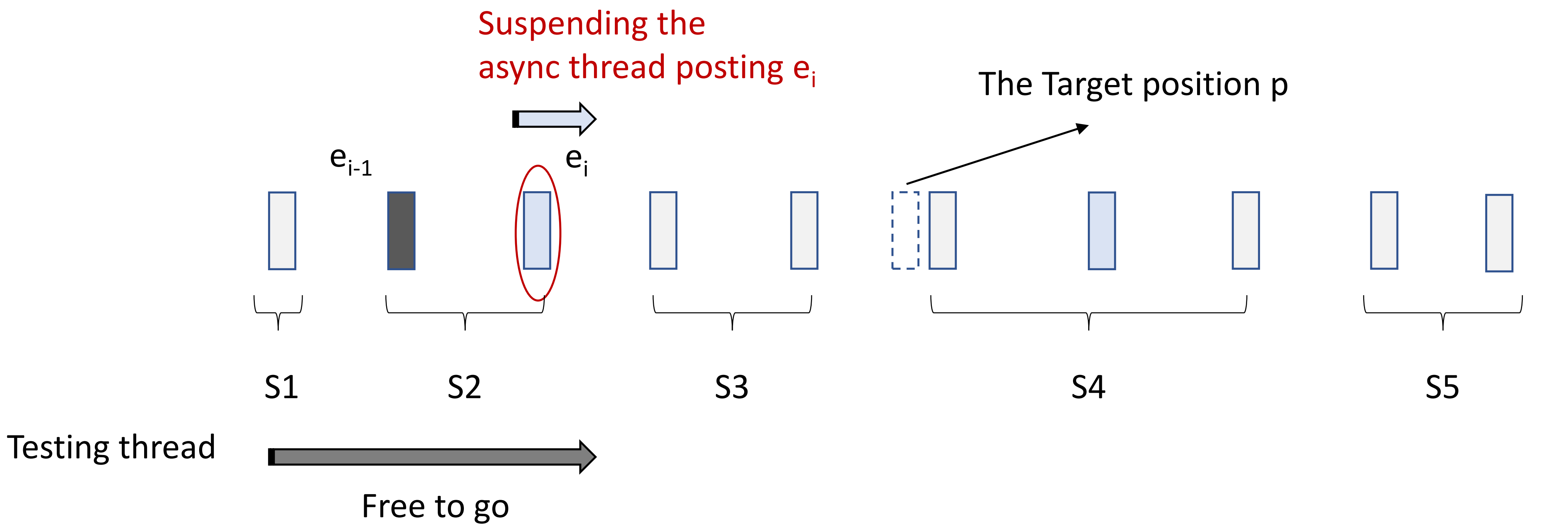
Identifying Schedule Space via Dynamic Analysis

- Scheduling an async event in statement boundary positions

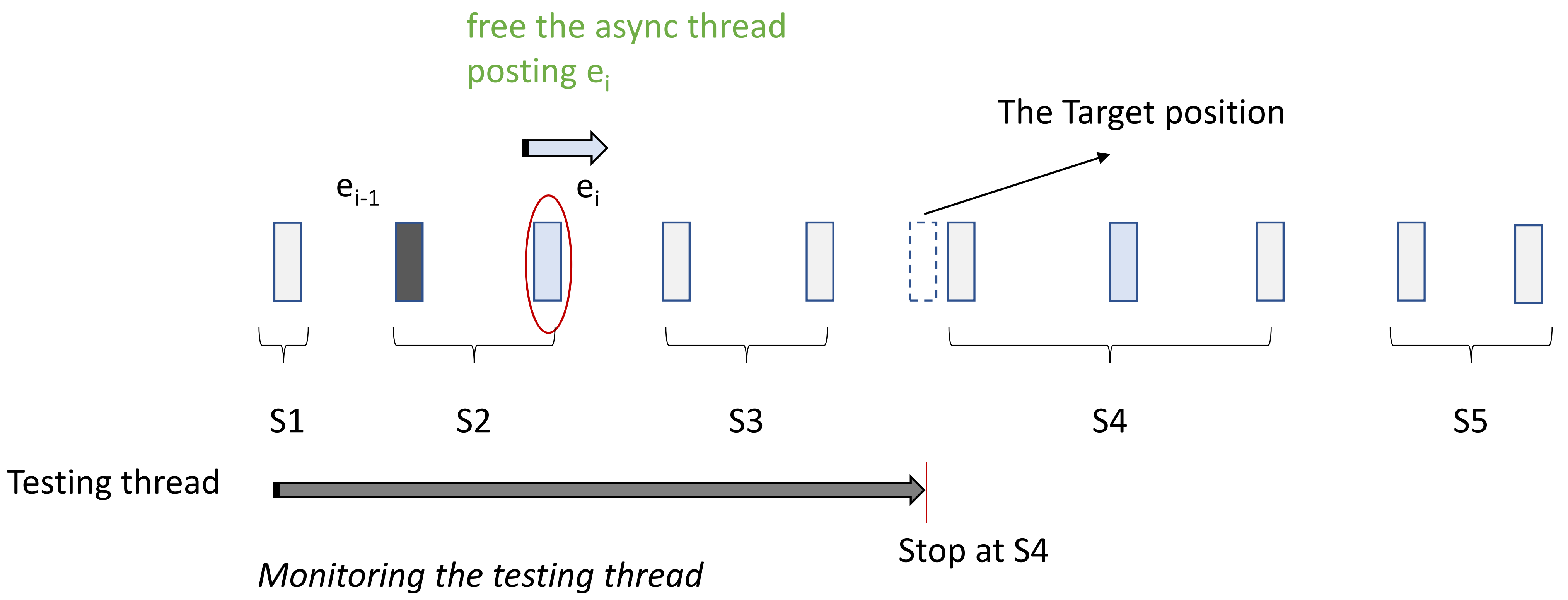


- Reason: more likely to trigger flaky test failures in those positions

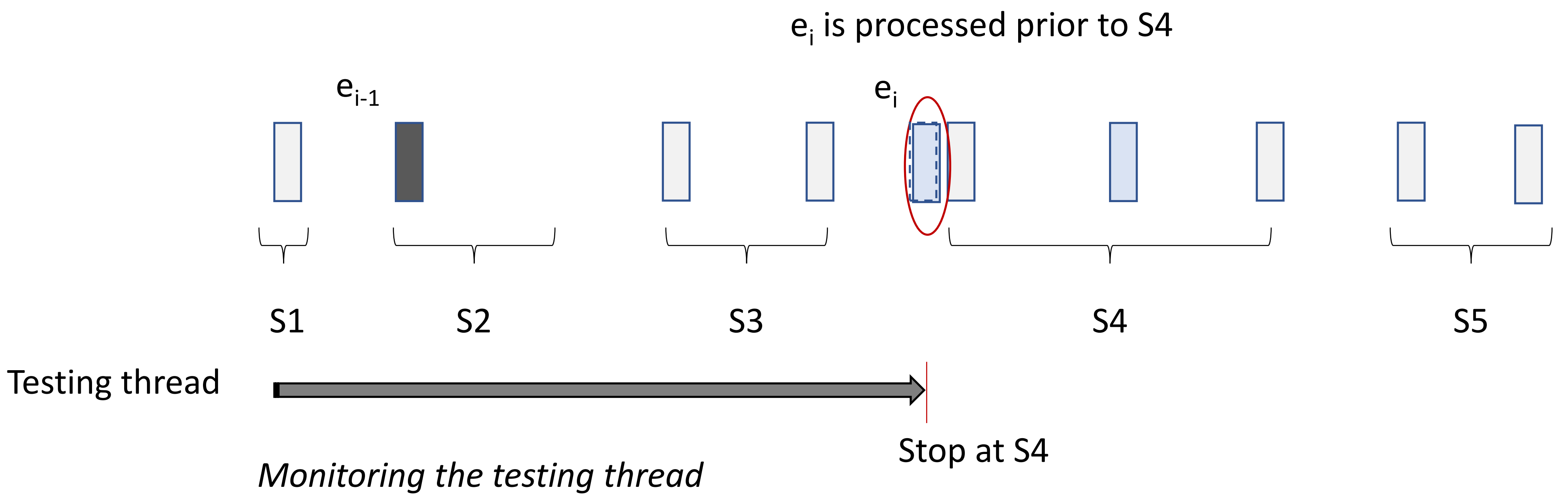
Scheduling An Event via Thread Operations



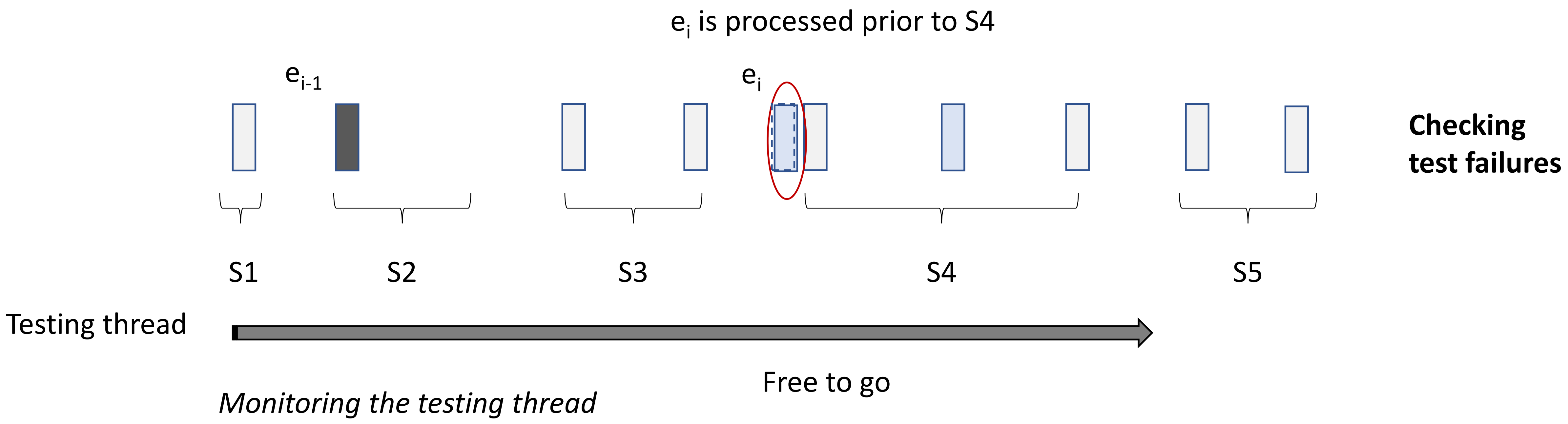
Scheduling An Event via Thread Operations



Scheduling An Event via Thread Operations

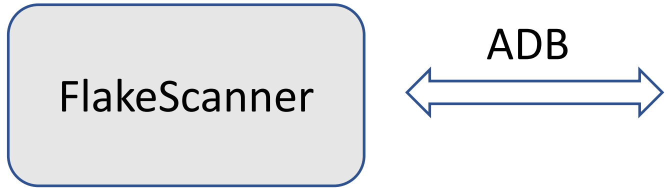

e_i is processed prior to S4
e_{i-1}
e_i
S1
S2
S3
S4
S5
Testing thread
Stop at S4
Monitoring the testing thread

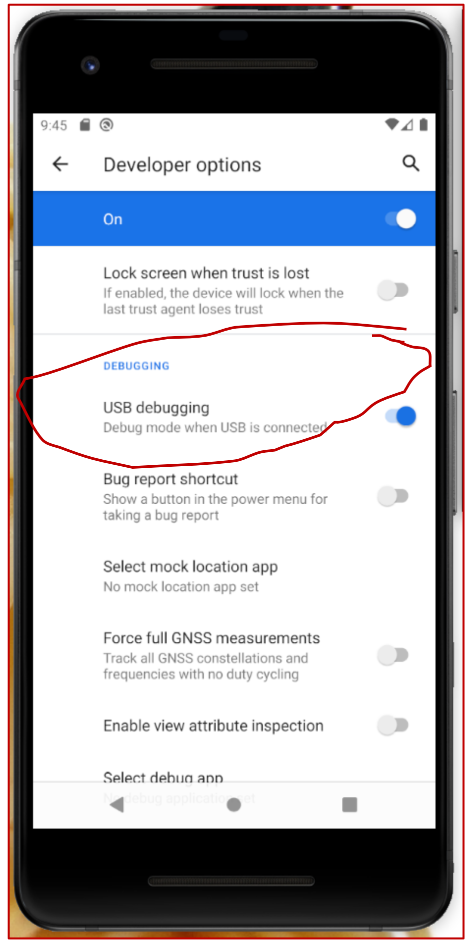
Scheduling An Event via Thread Operations



Implementation: FlakeScanner

- Android debugging tool (DDMS)
  - Hooking events
  - Operating threads





9:45

Developer options

On

Lock screen when trust is lost
If enabled, the device will lock when the last trust agent loses trust

DEBUGGING

USB debugging
Debug mode when USB is connected

Bug report shortcut
Show a button in the power menu for taking a bug report

Select mock location app
No mock location app set

Force full GNSS measurements
Track all GNSS constellations and frequencies with no duty cycling

Enable view attribute inspection

Select debug app

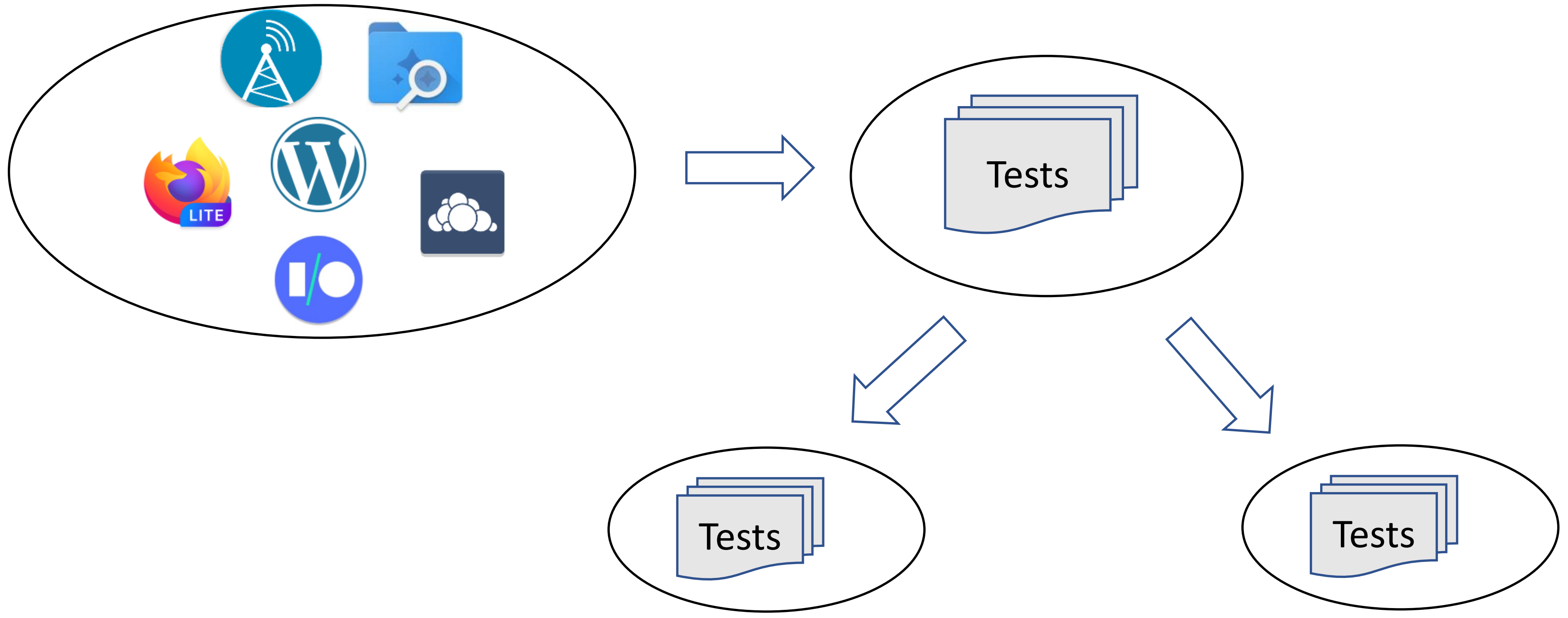
Evaluation

- Effectiveness on known flaky tests reported by developers?
- Comparing with existing techniques?
- Detecting unknown flaky tests in Android apps?

Subjects

33 Android projects,
2300 Github stars on average

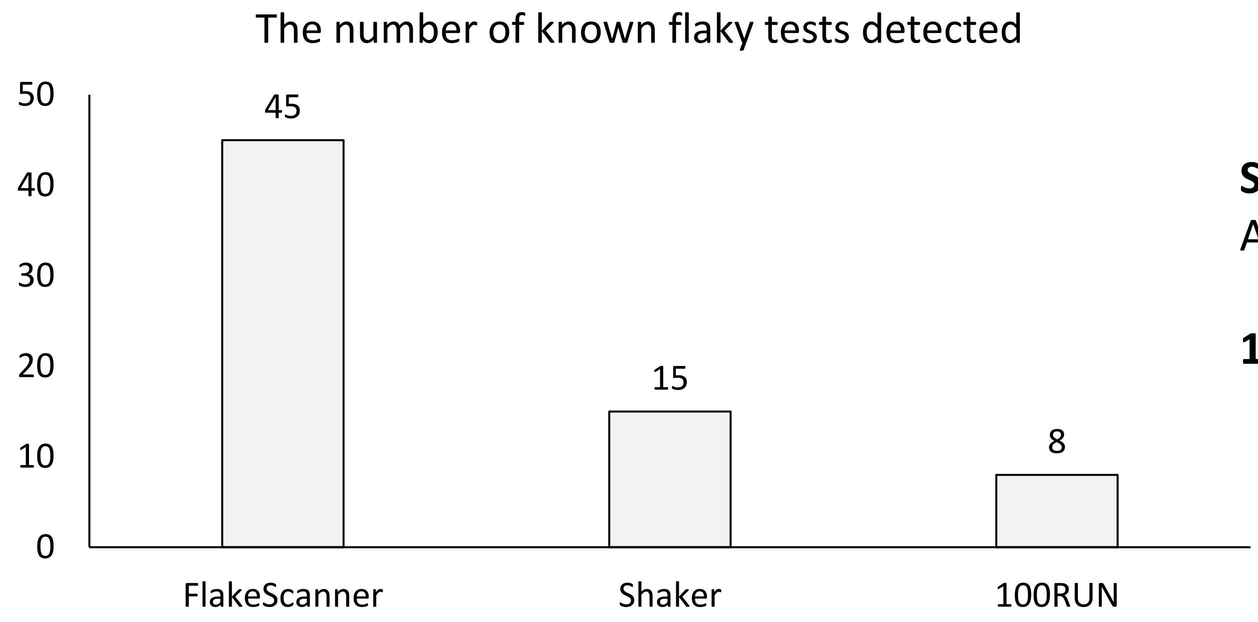
5000+ developer tests



Selected 52 known flaky tests

1444 tests (unknown)

Results: Effectiveness & Comparison



Shaker: a tool for detecting flaky tests in Android apps published on ICSME2020.

100RUN: running a test for 100 times.

Results on the 52 known flaky tests

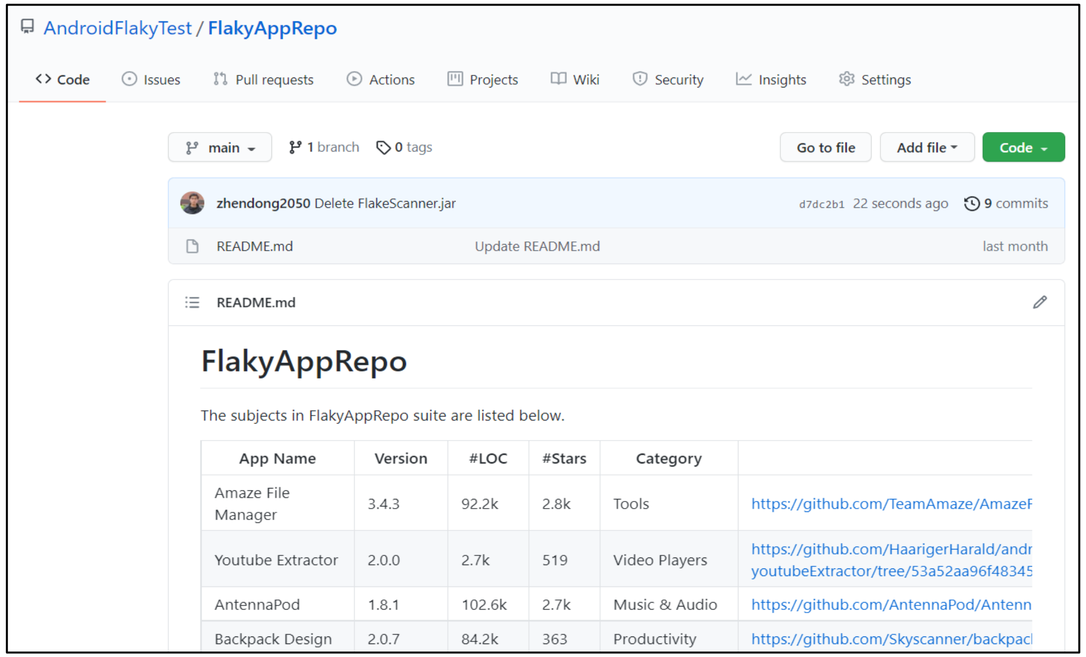
Results: Detecting unknown Flaky Tests

- Detected 245 flaky tests out of 1444 developer tests

- 13 got confirmed out of the reported 20 unknown flaky tests

Open Source of our tool and data set

- https://github.com/AndroidFlakyTest



AndroidFlakyTest / FlakyAppRepo

Code | Issues | Pull requests | Actions | Projects | Wiki | Security | Insights | Settings

main · 1 branch · 0 tags · Go to file · Add file · Code

zhendong2050 Delete FlakeScanner.jar d7dc2b1 22 seconds ago 9 commits

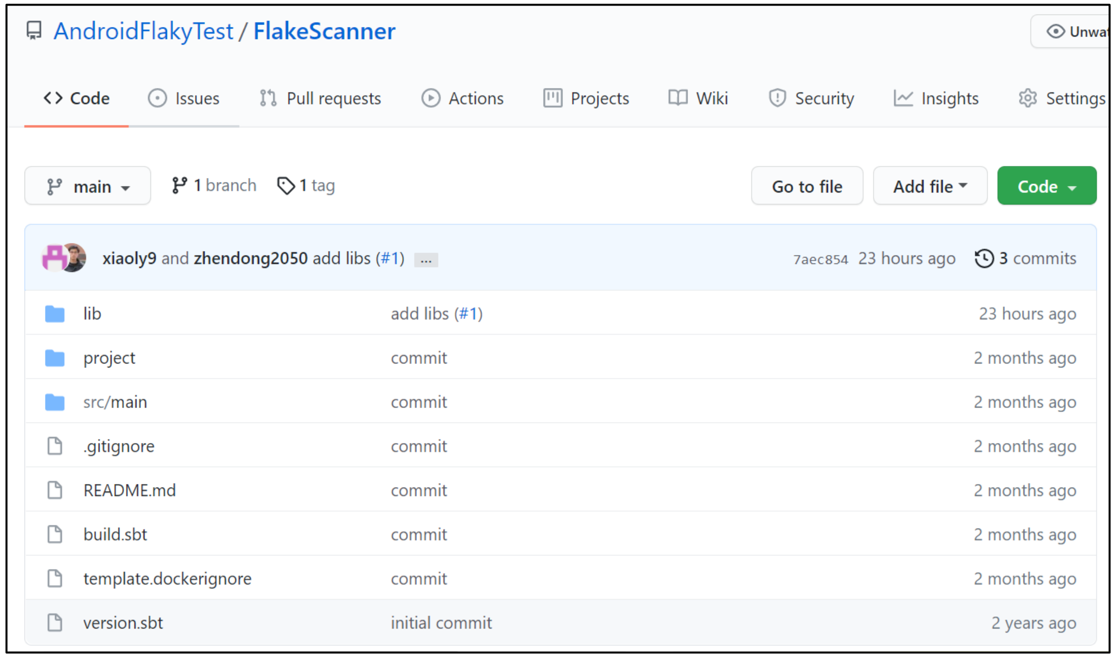
README.md	Update README.md	last month

README.md

FlakyAppRepo

The subjects in FlakyAppRepo suite are listed below.

App Name	Version	#LOC	#Stars	Category	
Amaze File Manager	3.4.3	92.2k	2.8k	Tools	https://github.com/TeamAmaze/AmazeF
Youtube Extractor	2.0.0	2.7k	519	Video Players	https://github.com/HaarigerHarald/andr youtubeExtractor/tree/53a52aa96f48345
AntennaPod	1.8.1	102.6k	2.7k	Music & Audio	https://github.com/AntennaPod/Antenn
Backpack Design	2.0.7	84.2k	363	Productivity	https://github.com/Skyscanner/backpac



AndroidFlakyTest / FlakeScanner

Code | Issues | Pull requests | Actions | Projects | Wiki | Security | Insights | Settings

main · 1 branch · 1 tag · Go to file · Add file · Code

xiaoly9 and zhendong2050 add libs (#1) 7aec854 23 hours ago 3 commits

lib	add libs (#1)	23 hours ago
project	commit	2 months ago
src/main	commit	2 months ago
.gitignore	commit	2 months ago
README.md	commit	2 months ago
build.sbt	commit	2 months ago
template.dockerignore	commit	2 months ago
version.sbt	initial commit	2 years ago